SE499 Report — Path Following Controllers

Rollen S. D'Souza
Software Engineering Undergraduate
Department of Electrical & Computer Engineering
`rs2dsouz@edu.uwaterloo.ca`

Acknowledgments

The author thanks Professor Christopher Nielsen for his guidance in developing the required intuition and mathematical tools for path following control design. Plots and simulations were partly written in Mathworks MATLAB under a student licence. Other simulations were written using C++ linked with the Boost library and Catch test framework.

Rollen S. D'Souza
Software Engineering Undergraduate
University of Waterloo
`rs2dsouz@edu.uwaterloo.ca`

Contents

Nomenclature

$\mathcal{T}$	Mobile robot task space.
$\mathcal{T}_{(w,h)}$	Rectangular mobile robot task space with width w and height h.
$\mathbf{M}$	A matrix. Can be assumed real unless stated otherwise.
$\mathbf{M}_{(i,j)}$	The value at the i-th row and j-th column of matrix $\mathbf{M}$. The variables may have a : substituted to indicate selection of all values in that dimension, similar to that found in the Matlab grammar.
$\mathbf{x}$	A vector in $\mathbb{R}$.
$\mathbf{x}_i$	The i-th component of vector $\mathbf{x}$

Chapter 1

Introduction

1.1 Background

A common control objective for mobile robots involves tracking a trajectory in the space the robot operates in. This space is defined as the task space of the robot, denoted as $\mathcal{T}$. Often the structure of this space is unknown — along with the goal — and the robot must first explore the world using an exploration and mapping algorithm before deciding on a goal and path. This paper is instead concerned with a restricted subset of this problem.

Consider a differential drive robot starting at location $(0, 0)$, placed in a world with an unknown number of obstacles, that is given the objective to reach position (x, y) in task space without colliding with obstacles. A two-pronged approach can be taken to safe-guard from collisions:

1. Plan a path that does not intersect with any obstacles for all future time.
2. Design a controller that ensures sufficiently fast convergence to the path and provides a guarantee, under reasonable assumptions, that the robot will not leave the path.

TODO : TALK ABOUT PATH PLANNING HISTORY AND STUFF
TODO : TALK ABOUT PROBLEMS WITH TRADITIONAL CONTROL METHODOLOGIES

1.2 Notation and Report Organization

This report assumes proficiency in multi-variable calculus and basic control theory. As a result, any results taken from theory of dynamical systems and other advanced undergraduate mathematical courses are stated and cited from a source used by the author. Scalars are denoted as x, vectors as $\mathbf{x}$ and matrices as $\mathbf{X}$. Time derivatives are denoted using Newton's dot notation and any other derivatives are denoted using the Leibniz notation.

In practice, planning algorithms generally precede control. However this report instead defers discussion of planning until after describing control methodologies in Section 2.1. This content placement is intended to give the reader a better intuition for the decisions made at planning stage.

1.3 Problem Statement

Let the task space, without loss of generality, be the rectangular space $\mathcal{T}_{(w,h)} = \{(x, y) \in \mathbb{R}^2 : 0 \le x \le w \wedge 0 \le y \le h\}$. For simplicity, consider the kinematic model of a differential drive robot with the combined position and orientation state vector $\mathbf{x} = (x, y, \theta)^\intercal$. The robot has a, possibly controllable, forward velocity $v : (\mathbf{x}, t) \mapsto \mathbb{R}$

with $v > 0$ and a controllable turning rate $u : (\mathbf{x}, t) \mapsto \mathbb{R}$. Unless stated otherwise, v is assumed constant. The kinematic dynamics follow as,

$$\dot{\mathbf{x}} = \begin{bmatrix} v\ cos(\mathbf{x}_3) \\ v\ sin(\mathbf{x}_3) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \tag{1.1}$$

The robot, at some unknown finite time, must arrive at the goal region $\mathcal{G} \subset \mathcal{T}$. Note there is no restriction in how it arrives in this region other than it does so safely. Starting at time $t = 0$ and then at a regular time interval Δ_p, the robot observes the world and decides, based on the observation, whether a new plan is required to approach the goal region. This may occur, for example, when the robot observes a new obstacle that the robot's currently planned path intersects. The simplest and most widely used planning algorithms are involve the generation of a tree that explores the current model of the world. This family of algorithms generate a sequence of waypoints. This is then used to generate a path $\sigma : \lambda \mapsto \mathcal{T}$ where λ is an arbitrary parameterization of the curve. Without loss of generality, this report assumes $\lambda \in [0, 1]$.

The path σ_1 is fed into the path following control which guides the robot over time towards, and along, the path in order to arrive in $\mathcal{G}$. Further, the control algorith must converge towards the path in a manner that is both guaranteed and as fast as possible. It must also faithfully follow the path, in order to reduce the chances of replanning. Given that the robot may not have a perfect model of the world, i.e. is not aware of all obstacles, a replan may still occur. The planning algorithm is required to design the new σ_2 so as to preserve any properties the control algorithm requires to stay on the trajectory continuously.

1.4 Description of Case Study

TODO : TALK ABOUT THE WATERLOO CAMPUS SIMULATION CASE STUDY

Chapter 2

Approach Description

2.1 Control

Three control strategies are considered for path following. The simplest controller, in terms of design and implementation, is the point-chasing controller. The other two techniques rely on the approach of transverse feedback linearization which decomposes the path following problem into transverse and tangential dynamical systems.

2.1.1 Point-Chasing Controller

A point-chasing controller involves simply treating the parameter λ of the path σ as a function t and designing a controller that converges to this moving point. In order to improve performance of this strategy, the control is designed around a point that leads the robots current position. That is, consider an observation $\mathbf{y}_l(\mathbf{x}(t)) = (\mathbf{x}_1, \mathbf{x}_2)^\intercal + l(cos(\mathbf{x}_3), sin(\mathbf{x}_3))^\intercal$ for some $l \in \mathbb{R}$, $l > 0$. We let $\mathbf{y}_{ref}(t) = \sigma(\lambda) = \sigma(t)$ where, for simplicity, we assumed that $\lambda = t$.

TODO : INSERT DERIVATION OF POINT CHASING CONTROLLER (TRACKING)

2.1.2 Transverse Feedback Linearization

General Form

Sylvester Approach

Serret-Frenet Frame Approach

2.2 Planning

2.2.1 RRT

2.2.2 Polynomial Splining

Essential Splining

Constrained Splining

Chapter 3

Case Study